



第三次上机作业(数组)求解思路

说明:

- 作业可用多种方法求解, 可使用本文档思路, 也可按自己的想法设计程序;
- 若程序中有逻辑问题(运行结果与预想不同), 可使用“单步调试”查找具体出现逻辑问题的语句, 详见“超星学习通”1.2节视频最后部分。

注:  表示屏幕输出  表示键盘输入

基础编程题:

1. 排序: 使用冒法和选择法对 10 个整数实现递减排序。

注:

- 10 个整数可以通过键盘输入或初始化时复制;
- 要求输出: 原始数组, 经过排序的数组, 以及冒泡法和选择法中变量交换的次数。

提示: 仿照课件的升序排序算法, 加入交换次数计数。

2. 对题 1 中得到的降序数组, 将数组的平均数插入数组中, 使更新后数组仍按照降序排列。

提示: 拷贝上题源代码, 做如下修改——

(1) 由于数组中需加入平均值, 因此数组长度改为 $N+1$;

(2) 在上题代码后, 添加循环计算数组元素之和, 得到平均值 `ave`;

(3) 使用循环, 从数组中下标为 0 的位置向后找 `ave` 插入的位置(比较), 找到后 `break` 跳出循环;

(4) 使用循环, 将下标为 $N-1$ 到“插入位置”的元素依次在数组中后移(即先移动原数组中最后一个元素、最后移动“插入位置”的元素以防止覆盖), 再将 `ave` 插入。输出更新后数组元素。

3. 统计大写辅音字母个数: 某文本共有 3 行, 每行不超过 80 个字符, 试编写一个程序统计该文本中大写辅音字母的个数(文本由键盘输入)。

例: 输入 





输出 


提示:

(1) 定义 3 行 80 列的二维字符数组;

(2) 对每个字符串, 从第一个字符开始到 `\0`, 判断各字符是否为大写并且不等于 `'A'`、`'E'`、`'I'`、`'O'`、`'U'`, 若满足条件, 则计数加 1。

4. 字符串连接: 试编写一个程序连接两个字符串(不使用库函数 `strcat()`), 连接后两个字符串之间加个空格。

例: 输入 



输出 

提示:

(1) 定义两个字符数组存放两个字符串;

(2) 找到第一个字符数组的 `\0`, 将该位置赋值为 `' '` (空格);

(3) 从第一个字符数组中添加 `' '` (空格) 的后一个下标位置开始, 将第二个字符数组中各个字符进行拷贝。

5. 矩阵运算：读入 1 个正整数 $n(1 \leq n \leq 6)$ ，再读入 n 阶方阵，计算该矩阵除副对角线、最后一列和最后一行以外的所有元素之和。

例：输入

4
2 3 4 1
5 6 1 1
7 1 8 1
1 1 1 1

输出 sum=35

提示：计算矩阵中元素之和，当取到矩阵副对角线(右上角到左下角的对角线)、最后一列($j=n-1$)和最后一行($i=n-1$)元素时，continue。

设计类编程题：

6. 统计：从键盘输入10个学生的数学(MT)、英语(EN)和物理(PH)成绩，并按照如下统计形式输出，包括学生学号(NO)、各科成绩、总成绩(SUM)、平均分(AVE)及是否每科都超过90分('Y' or 'N')

NO	MT	EN	PH	SUM	AVE	>90
1	97	87	92	276	92	N
2	92	91	90	273	91	Y
3	90	81	82	253	84	N

注：

- 每科成绩为整数。
- 只需要定义一个整形数组即可

提示：仿照课堂例题统计学生总成绩及平均成绩。

(1) 假设共有 N 个学生、 M 门课程，由于字符型与整型之间具有互通性，因此本题定义一个二维整型数组即可。二维数组的行数为 N (N 个学生)、列数为 $M+4$ (还需存储学号、总分、平均分、标志位)；

(2) 使用循环输入每个学生的 M 门课成绩，无需输入学号 (可在循环中自动赋值)；

(3) 统计学生成绩时，外层循环执行 N 次，实现对 N 个学生的统计。循环体中对第 i 的学生统计时，需首先置该生的总分为 0，再通过内层循环取得所有科成绩累加到总分。当 M 科成绩均大于 90 时，标志位赋值为 'Y'，否则赋值为 'N'。

(4) 以表格形式输出时，注意对齐。

7. 找出任意一个二维数组中的鞍点。

所谓鞍点，即是该位置上的元素在所在行中最大而在所在列中最小。一个二维数组中可能有多个鞍点，也可能没有鞍点。要求输出所有鞍点的位置，无鞍点时给出有关信息。

例：

<pre>Input array: 1 2 3 4 5 2 4 6 8 10 3 6 9 12 15 4 8 12 16 20 The saddle point of this matrix is (0,4):5 Press any key to continue</pre>	<pre>Input array: 1 2 3 4 11 2 4 6 8 12 3 6 9 10 15 4 8 12 16 7 This matrix have no saddle point. Press any key to continue</pre>
--	---

```

Input array:
6 6 6
6 6 6
The saddle point of this matrix is (0,0):6
The saddle point of this matrix is (0,1):6
The saddle point of this matrix is (0,2):6
The saddle point of this matrix is (1,0):6
The saddle point of this matrix is (1,1):6
The saddle point of this matrix is (1,2):6
Press any key to continue

```

提示：对于 N 行 M 列的二维数组——

- (1) 定义 `int a[N][M]`，以及 `int row[N]` 存放各行最大值、`int col[M]` 存放各列最小值，标志位 `flag=0`（初始值 0 表示 `a` 中无鞍点）；
- (2) 使用嵌套循环，从键盘输入二维数组中元素；
- (3) 分别使用循环设置数组 `row[]` 和 `col[]` 中的值；
- (4) 使用嵌套循环，利用数组 `row` 和 `col`，判断 `a` 中每个元素是否为鞍点，若某元素为鞍点，则输出鞍点信息（位置及值），置 `flag=1`；
- (5) 循环后，若 `flag` 为 0，输出无鞍点信息。

思考题（选做）

8. 十六进制转换十进制

(1) 初级进制转换问题：输入一个字符串（字符串的长度不超过 7，不包括 '\0'），对其做如下处理：滤去所有的非十六进制字符后，组成一个新字符串（十六进制形式），然后将其转换为十进制数后输出。

例： 输入： **Ha4-1**

输出： **十六进制：0xA41 十进制：2625**

提示：

(1) 定义 `char str[8]`，输入字符串长度不超过 7，即对应的最大十六进制字符串为“FFFFFFF”，对应的十进制数为 268435455，在 `long` 型数据的表示范围之内（-2147483648~2147483648），因此本题定义 `long x=0`；即可保存十进制结果；

(2) 由于输入字符串中可能包含空格，因此需使用 `gets()` 函数；

(3) 使用循环，从字符串下标为 0 的位置开始到 '\0'，逐个取出字符 `str[i]` 并判断 `str[i]` 是否在 '0'~'9' 或 'a'~'f' 或 'A'~'F' 的范围内，若在上述范围内则保留在 `str[]` 中（用 `str[i]` 为 `str` 数组中下标为 `j` 的位置赋值，初始 `j=0`，条件为真则 `str[j++]=str[i]`；覆盖原字符），循环中还需将小写 'a'~'f' 转换为 'A'~'F'（题目输出要求）。循环结束后，为 `str[j]` 赋值结束符 '\0'。

滤去非十六进制字符后，例中对应的 `str[]` 中为 "A41"；

(4) 十六进制转十进制规则：从 `str[]` 中逐个取字符直到 '\0'（循环），`x=x*16+str[i]` 对应的数字，即若 `str[i]` 在 '0'~'9' 范围内则 `x=x*16+str[i]-'0'`；若 `str[i]` 在 'A'~'F' 范围内则 `x=x*16+str[i]-'A'`；因此，`x` 的初始值必须为 0；

如十六进制 A41 对应的十进制 $=((0*16+10)*16+4)*16+1=2625$ ，其中 0 为 `x` 的初始值，之后逐个从 `str[]` 中取出 'A'（对应十进制 10）、'4'、'1'，直到 `str[]` 的 '\0'；

(5) 按要求格式输出。

(2) 高级进制转换问题：输入的字符串长度不超过 20（不包括 '\0'），对其做如下处理：滤去所有的非十六进制字符后，组成一个新字符串（十六进制形式），然后将其转换为十进制数后输出。

例： 输入： **A4 1cDb ACDB**

输出： **十六进制：0xA41CDBACDB 十进制：704858795227**

提示：长整数需要使用数组进行存储。

提示:

(1) 定义 `char str[21]`, 输入字符串长度不超过 20, 则对应的最大十六进制字符串的十进制超出了 `long` 型数据的表示范围, 因此本题需要定义整型数组存储长整型数的各位。

定义 `int start[100]={0}, ans[100]={0}, res[100]={0}`; 存储被除数、商、余数

最终十进制结果(逆序)保存在 `res[]` 中, 如例, `res[]` 保存的数值为 {7,2,2,5,9,7,8,5,8,4,0,7};

(2) 同上一小题;

(3) 同上一小题

滤去非十六进制字符后, 例中对应的 `str` 中为 "A41CDBACDB";

(4) 使用循环, 将 `str[]` 中的每位十六进制字符转为对应十进制, 存放在 `start[]` 中 (从下标为 1 的位置存放), `start[0]` 存放十六进制数的位数。例中对应的 `start[0]` 为 10, `start[1]~start[10]` 分别保存 10,4,1,12,13,11,10,12,13,11;

(5) 大数十六进制转十进制规则: 可百度“浅谈大数的进制转换”, 注意最先得到的是转换后十进制数的最后一位, 因此 `res[]` 中存放的数据为逆序;

(6) 按要求格式输出。